Model Tuning

BA810: Supervised Machine Learning

Nachiketa Sahoo

Summary of Mid-course Feedback

Continue

1. Labs, more time, in detail
2. Recap (with questions)
3. Interactive class, Q&A

Stop

1. Too much Q&A!
    1. Delaying labs
2. Assuming advanced statistics background

Start

1. More examples and illustrations in lecture
2. In-class lab exercises
3. Video resources
4. Moving Q&A offline

Course Map (Topics)

1. Introduction to Machine Learning
2. General predictive models
   - Regression and classification
3. Model tuning and selection
4. End-to-end Machine Learning process
5. Specific predictive models
   - Support Vector Machines, Decision Tree, Ensembles
6. Managing class imbalance in practice

Recap

- Cost sensitive classification
  - Using cost/benefit to choose models
  - Optimize prediction threshold to minimize cost

- Validation and Cross-validation
  - Estimate test-error or generalization error
  - Validation: split data in two (train on one part, predict on the other)
    - Unreliable estimation; significantly overestimates the test error
  - Cross-validation: split into K parts (test on k’th, train on rest, K times)
    - Error on all records are averaged → more reliable
    - K = 5–10, typically works well

Recap/Follow-up

- Regularization: optimize objective that penalizes large parameters
  - Ridge: $MSE + \lambda\sum\beta_p^2$: reduces variance, shrinks but doesn't turn any β_p off
  - Lasso: $MSE + \lambda\sum|\beta_p|$: reduces variance, shrinks *and turns some β_p to zero*

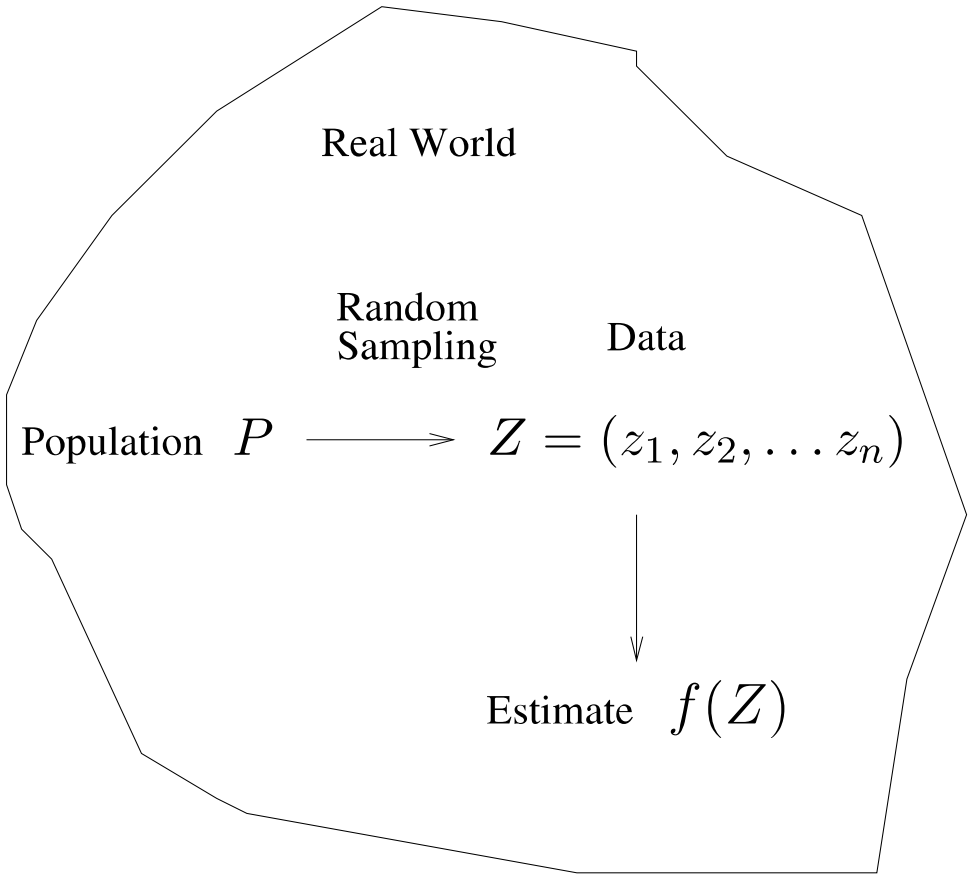
- For each of a set of λs
  - Divide data into K-folds
    - Keep each fold in turn for validation, use the rest for training using the chosen λ
  - Average the K prediction-errors (cross-validation error)
- Choose the λ with the lowest cross-validation error
  - Retrain using this λ on the entire data for a model to deploy
- The cross-validation error isn't an accurate estimation of test/generalization-error
  - Selected λ based on entire dataset; nothing left for an honest generalization error estimation
  - If interested in generalization error, set aside a test data before CV

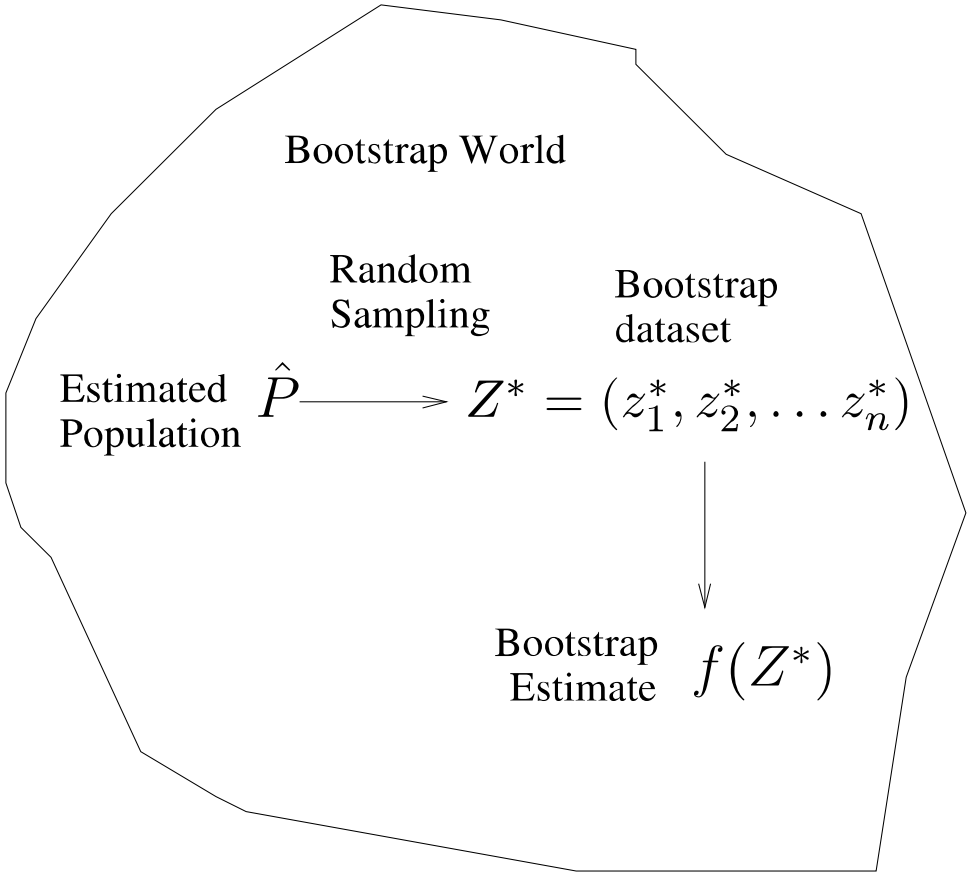
Bootstrap

- An easy way to quantify uncertainty with an estimate
  - Parameter value (statistically significantly different than 0?)
  - Prediction error/accuracy, etc. (are the performances of two models significantly different?)
- If we have a dataset of size n, ideal process
  - Draw B random datasets of size n from the original source
  - Estimate B times, measure the standard deviation (of regression coefficient or prediction accuracy)
- Real-world gives us only one dataset
  - Draw B random datasets of size n from the dataset we have with replacement
  - Estimate B times, measure the standard deviation

  (bootstrap ≈ make it work with what you have, without outside data help)

Bootstrap





(source: Trevor Hastie and Rob Tibshirani, https://www.dataschool.io/)

Grid search vs Random search

Techniques for Selecting Hyperparameters (k and type of distance for kNN; λ for Ridge/Lasso regression; etc.)

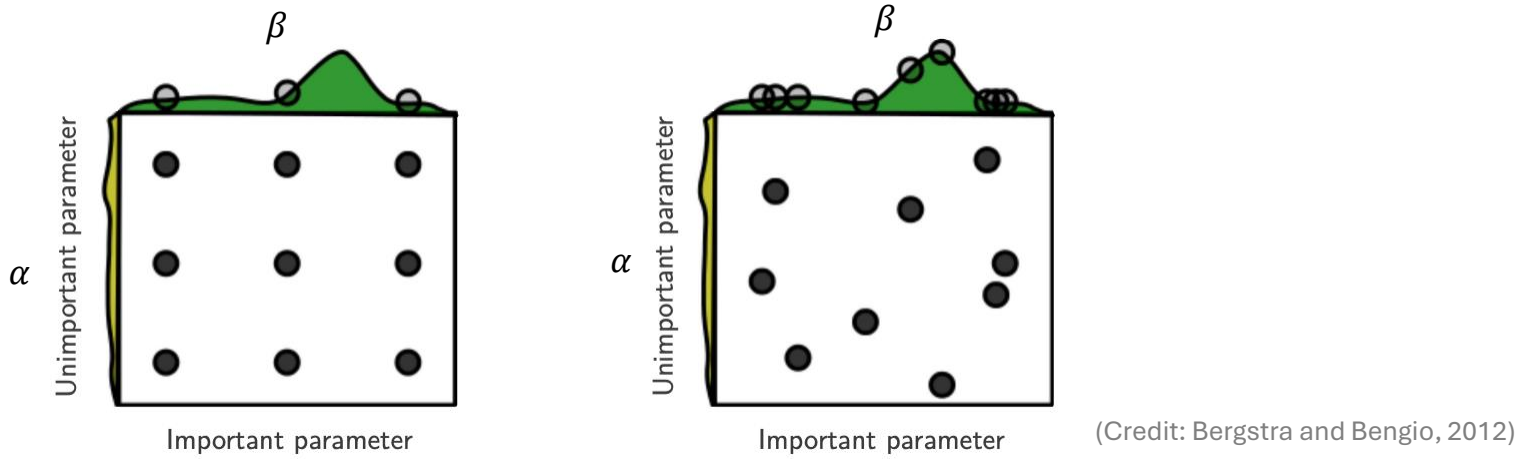
Grid Search

- Define a Model/Pipeline
- Specify a "hyperparameter grid"
  - Consider all combination of parameter values, exhaustively
- For each combination
  - Evaluate the prediction performance of the model using CV

Random Search

- Define a Model/Pipeline
- Specify a "hyperparameter distributions"
  - For a pre-specified number of times, draw a value from each
- For each combination
  - Evaluate the prediction performance of the model using CV

Grid search vs Random search



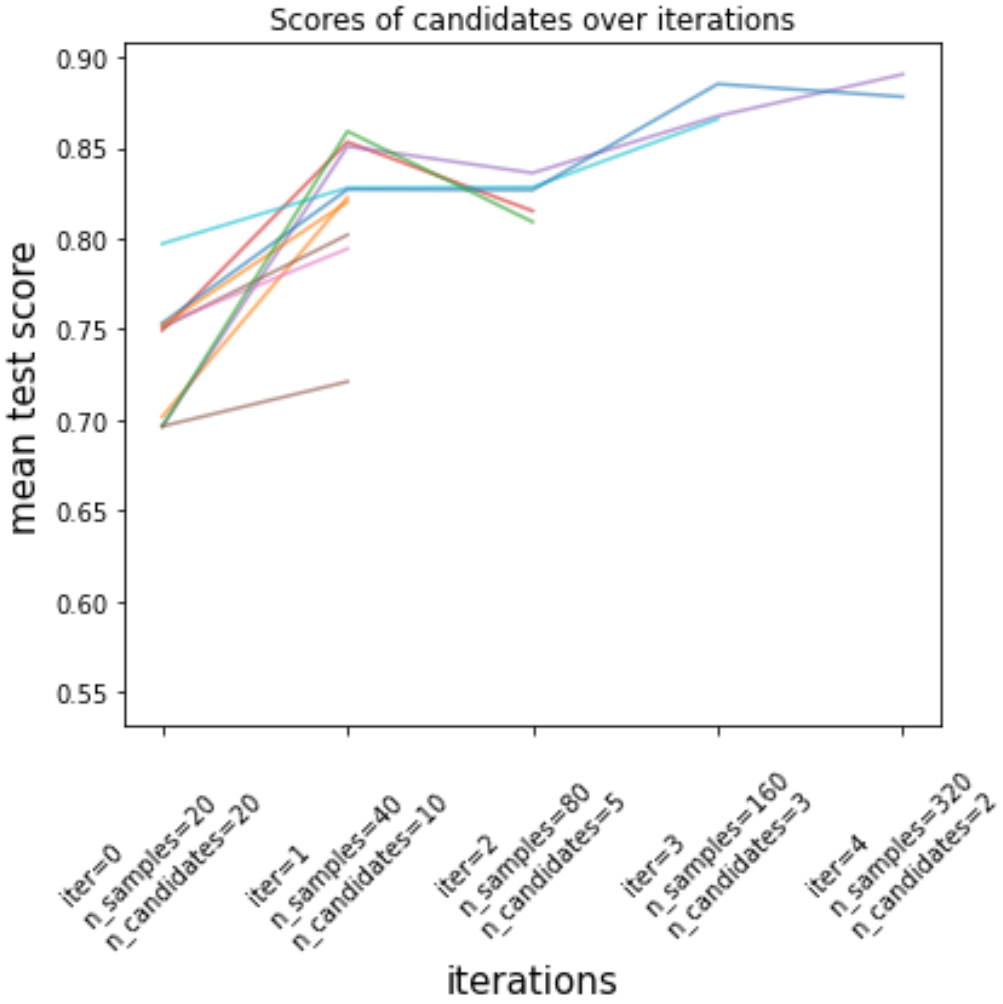
(Credit: Bergstra and Bengio, 2012)

- Model Performance varies by two hyper-parameters — α and β — as shown on the margin
  - More sensitive to β (but we won't know that until testing)
- Grid search explores only one dimension at a time, keeping the other(s) fixed
  - Random search explores different values of α and β in each draw, thus can evaluate more unique values of each hyperparameter for the same number of evaluations.
  - For high dimensional problems Random search is more likely to find good hyper parameter values

Random/Grid Search with Successive Halving/Reduction

1. Carry out Grid or Random search as before, but with limited r "resources"
   E.g., 20-50 records (data samples)
2. Pick top $\frac{1}{f}$ parameter combinations (e.g., $f = 2,3$), increase resource (set $\mathrm{r} = r \times f$)
3. Repeat 1–2 until only one parameter combination remains.

Explore same number of parameters as before, but more precisely evaluate the promising ones.



Summary

- Bootstrap to measure uncertainty
  - Randomly sample equal sized data with replacement
  - Repeat estimation and accuracy measurement
- Hyper-parameter search
  - Grid: explore all combinations of fixed hyper-parameter values
  - Random: Evaluate combination of one random draw for each hyperparameter; repeat pre-defined number of times
  - Halving Grid/Random: evaluate hyper-parameter values identified as before, but all only cheaply; promising ones more precisely

Next class

- End-to-end ML example
- Complete mid-course team feedback at https://teamlearning.bu.edu/ by Thu (11/9) noon